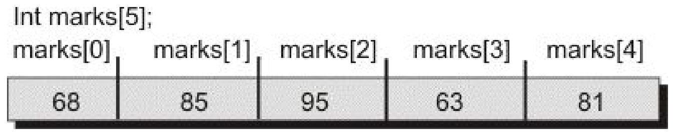
Unit-6

Array

Introduction

Array is the collection of similar data types that can be represent by a single variable name. The individual data items in an array are called elements. A data items is stored in memory in one or more adjacent storage location dependent upon its type. The data types may be any valid data types like char, int or float. The element of array share same variable name but each element has a different index of number known as subscript. In c subscript starts from zero, here if age [10] is the variable name so the first element will be age [0] and second age [1] and so on. Array can be single dimensional or multidimensional the array of subscript determines the dimension of array. One dimensional has one subscript and two dimensional has two subscript and so on.

Example: int marks[5] = {68, 85, 95, 63, 81};



Types of Array

A) One dimensional array

<u>Declaration of one dimensional array</u>

Like as the variable array should be declared before they are used in program.

The syntax of declaration of an array

data_type array_name [size];

Here the data type may be any elements of c data type set. The array_name is the name of array used to stored data. The size of array specifies the number of elements that can be stored in the array. The size of the array should be always positive integer.

int age[10];

float salary[20]

Here the element stored in array variable are age[0],age[1],age[2],age[3],……….. age[8], age[9]

same as the salary are salary[0], salary[1], salary[2], salary[3], salary[04]…………. salary[19].

Accessing 1-D array Elements

Once the array is declared, individual elements are referred. This is done with the subscript or index which is the integer in brackets followed by the array name. This number referring to an array element this number specifies the position of the element in the array. Whereas while declaring array the number specifies the size of the array i.e. number of element in that array. The array index can range from 0 to size-1.

In an array if marks [10] then the terms of array are

Lower bound=0

Upper bound =size-1=10-1=9

Size =upper bound+1=10

Processing array element

For processing the arrays we generally use loop and the loop variable is used at the place of subscript. The initial value of loop variable is taken 0 since array subscript start from zero.

Initialization of one dimensional array

We can explicitly initialize arrays at the time of declaration. The syntax for initialization of an array is

data_type   array_name[size]={value1,value2,…………valueN};

Here array_name is the name of the array variable, size is the size of the array and value1, value2,…….valueN are the constant values which are assigned to the array elements one after another.

Example.

int marks[5]={55,59,78,87,65};

The values of the array elements after this initialization are

marks[0]=55, marks[1]=59, marks[2]=78, marks[3]=87, marks[4]=65

Exercise:

1) Write a program to read 10 numbers from user and display it on the screen.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int arr[10],i;
clrscr();
for(i=0;i<10;i++)
{
printf("Enter elements of array\n");
scanf("%d",&arr[i]);
}
printf("Displaying array elements\n");
for(i=0;i<10;i++)
{
printf("%d\n",arr[i]);
}
getch();
}
```

2) Write a program to display smallest and largest element from N number of array elements.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int arr[20],i,n,small,large;
```

```c
clrscr();
printf("How many number of elements are there in array\n");
scanf("%d",&n);
for(i=0;i<n;i++)
{
printf("Enter elements of array\n");
scanf("%d",&arr[i]);
}
small=arr[0];
large=arr[0];
for(i=0;i<n;i++)
{
if(small>arr[i])
small=arr[i];
if(large<arr[i])
large=arr[i];
}
printf("Smallest element =%d and largest element =%d",small,large);
getch();
}
```

3) Write a program to search an element from N number of array elements.

```c
#include<stdio.h>
#include<conio.h>
void main()
{
int arr[20],search,i,n;
clrscr();
printf("How many number of elements are there in array\n");
scanf("%d",&n);
for(i=0;i<n;i++)
{
printf("Enter elements of array\n");
scanf("%d",&arr[i]);
}
printf("Enter element to be searched\n");
scanf("%d",&search);
for(i=0;i<n;i++)
{
if(arr[i]==search)
{
printf("Search element found\n");
break;
}
}
if(i==n)
printf("Element not found\n");
getch();
}
```

4) Write a program to sort N numbers in ascending order (using bubble sort).

```c
#include<stdio.h>
#include<conio.h>
```

```
void main()
{
int arr[20],i,j,tem,n;
clrscr();
printf("How many elements are there in an array\n");
scanf("%d",&n);
for(i=0;i<n;i++)
{
printf("Enter elements\n");
scanf("%d",&arr[i]);
}
for(i=0;i<n-1;i++)
{
for(j=0;j<n-1-i;j++)
{
if(arr[j]>arr[j+1])
{
tem=arr[j];
arr[j]=arr[j+1];
arr[j+1]=tem;
}
}
}
printf("\nsorted elements are\n");
for(i=0;i<n;i++)
{
printf("%d\n",arr[i]);
}
getch();
}
```

5) Write a program to sort N numbers in ascending order (using selection sort).

```
#include<stdio.h>
#include<conio.h>
void main()
{
int arr[20],i,j,tem,n;
clrscr();
printf("How many elements are there in an array\n");
scanf("%d",&n);
for(i=0;i<n;i++)
{
printf("Enter elements\n");
scanf("%d",&arr[i]);
}
for(i=0;i<n-1;i++)
{
for(j=i+1;j<n;j++)
{
if(arr[i]>arr[j])
{
tem=arr[i];
arr[i]=arr[j];
arr[j]=tem;
}
}
}
printf("\nsorted elements are\n");
for(i=0;i<n;i++)
{
printf("%d\n",arr[i]);
}
getch();
}
```

B) Multidimensional Array

An array having more than one subscript is called multidimensional array. Two dimensional array have two subscript, three dimensional array have three subscript and so on.

Two dimensional arrays

Two dimensional array is declared as follows:

data_type array_name [firstsubscript][secondsubscript];

Here, first subscript represents number of rows in an array and second subscript represents number of columns in an array.

Initialization of two dimensional array

int a[2][3]={3,5,8,9,4,6};

Which means,

a[0][0]=3,   a[0][1]=5,   a[0][2]=8,   a[1][0]=9,   a[1][1]=4,   a[1][2]=6.

Processing two dimensional array

For processing 2-D array, we must use nested loop. The outer loop corresponds to the row and the inner loop corresponds to the column.

Exercise

1. Write a program to read any 2×2 matrix and display it in appropriate format.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[2][2],i,j;
clrscr();
for(i=0;i<2;i++)
{
for(j=0;j<2;j++)
{
```

```
printf("Enter elements of matrix\n");
scanf("%d",&a[i][j]);
}
}
for(i=0;i<2;i++)
{
for(j=0;j<2;j++)
{
printf("%d\t",a[i][j]);
}
printf("\n");
}
getch();
}
```

2. Write a program to display the sum of all elements of any 3×2 matrix.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[3][2],i,j,sum=0;
clrscr();
for(i=0;i<3;i++)
{
for(j=0;j<2;j++)
{
printf("Enter elements of matrix\n");
scanf("%d",&a[i][j]);
}
}
for(i=0;i<3;i++)
{
for(j=0;j<2;j++)
{
sum=sum+a[i][j];
}
}
printf("Sum of all elements of matrix =%d",sum);
getch();
}
```

3. Write a program to display sum of diagonal elements of any given 3×3 matrix.
4. Write a program to display sum of each rows of any given n×m matrix.
5. Write a program to display transpose of any given n×m matrix.
6. Write a program to display sum of each columns of any given n×m matrix.
7. Write a program to display the sum of any given two n×m matrix.

8. Write a program to display the product of any given two n×m matrix.

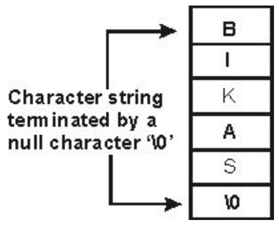
String

String is an array of characters or alphabets. Each programming language has a set of character to communicate with computers. These are:

(a) Alphabet: A, B, C,........ X, Y, Z and a, b, c.....x , y , z

(b) Digits: 1, 2, 3 .....9 .

(c) Special Characters: /, * , ( ) , = , ' , etc .



Some important facts about string:

(i) A finite set of zero or more characters is called – String.

(ii) Length: The number of characters including space is called size or length.

Example:" Hello Word!" – Its length is 11 including blank spaces.

(iii) A string constant is denoted by a characters included in double quotes.

(iv) The string array contains NULL character ('\0') to insure own existence to compiler.

Example:

"BIKAS"

"Microsoft Products"

" 521457 "

String Library Functions

There are several library functions used to manipulate strings. The prototypes for these functions are in header file string.h. We'll discuss some of them below-

strlen ( )

This function returns the length of the string i.e. the number of characters in the string excluding the terminating null character. It accepts a single argument, which is pointer to the first character of the string. For example strlen("suresh") returns the value of 6. Similarly if s1 is an array that contains the name "deepali" then strlen(s1) returns the value 7.

```
/*Program to understand the work of strlen() function*/

#include<stdio.h>

#include<string.h>

main( )

{

        char str[20];

        int length;

        printf("Enter the string:\n");

        scanf("%s", str);

        length = strlen(str);

        printf("Length of the string is : %d\n", length);
}
```

Output:

```
Enter the string : programming
Length of the string is : 11
```

strcmp ( )

This function is used for comparison of two strings. If the two strings match, strcmp( ) returns a value 0, otherwise it returns a non-zero value. This function compares the string character by character. The comparison stops when either the end of string is reached or the corresponding characters in the two strings are not same. The non zero value returned on mismatch is the difference of the ASCII values of the non matching characters of the two strings-

strcmp (s1, s2) returns a value-

<0 when s1 < s2

= 0 when s1 == s2

> 0 when s1 > s2

Generally we don't use the exact non zero value returned in case of mismatch. We only need to know its sign to compare the alphabetical positions of the two strings. We can use this function to sort the strings alphabetically.

```
/* Program to understand the work of strcmp( ) function*/
#include<stdio.h>
#include<string.h>
void main( )
{
    char str1[10], str2[10];
    printf("Enter the first string:");
    scanf("%s", str1);
    printf("Enter the second string:");
    scanf("%s", str2);
    if((strcmp(str1, str2)) == 0)
      printf("Strings are same\n");
    else
      printf("Strings are not same\n");
}
```

Output:

```
Enter the first string : Nepalgunj
Enter the second string : Gaindakot
```

```
Strings are not same
```

strcpy ( )

This function is used for copying one string to another string, strcpy(srt1, str2) copies str2 to str1. Here str2 is the source string and str1 is destination string. If str2 = “programming” then this function copies “programming” into str1. This function takes pointers to two strings as arguments and returns the pointer to first string.

```
/* Program to understand the work of strcpy( ) function*/
#include<stdio.h>
#include<string.h>
void main( )
{
    char str1[10], str2[10];
    printf(“Enter the first string:”);
    scanf(“%s”, str1);
    printf(“Enter the second string:”);
    scanf(“%s”, str2);
    strcpy(str1, str2);
    printf(“First string : %s \t\t Second string: %s\n”, str1, str2);
    strcpy(str1, “Jomsom”)
    strcpy(str2, “Kagbeni”)
    printf(“First string : %s \t\t Second string: %s\n”, str1, str2);
}
```

Output:

```
Enter the first string : Balaju
Enter the second string : Pokhara
First string : Pokhara              Second string : Balaju
First string : Jomsom               Second string : Kagbeni
```

The programmer should take care that the first string has enough space to hold the second string.

The function calls like strcpy(“New”, strl) or strcpy(“New”, “York”) are invalid because “New” is a string constant which is stored in read only memory and so we can’t overwrite it.

```
strcpy(“New”, str1);                /*Invalid*/
strcpy(“New”, “York”);              /*Invalid*/
```

strcat( )

This function is used for concatenation of two strings. If first string is “King” and second string is “size” then after using this function the first string becomes “Kingsize”.

```
strcat(str1, str2);                 /*concatenates str2 at the end of str1*/
```

The null character from the first string is removed, and the second string is added at the end of first string. The second string remains unaffected.

This function takes pointer to two strings as arguments and returns a pointer to the first(concatenated) string.

```
/* Program to understand the work of strcat( ) function*/

#include<stdio.h>

#include<string.h>

void main( )

{

        char str1[20], str2[20];

        printf("Enter the first string:");

        scanf("%s", str1);

        printf("Enter the second string:");

        scanf("%s", str2);

        strcat(str1, str2);

        printf("First string:%s \t Second string: %s\n", str1, str2);

        strcat(str1, "_one");

        printf("Now first string is : %s\n", str1);

}
```

Output:

```
        Enter the first string : data
        Enter the second string : base
        First string : database                                 Second string : base
        Now first string is : database_one
```

strrev( )

This function takes string as an argument and returns the string in reverse order.

strrev(string)

Example

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char str[10];
clrscr();
printf("Enter a string\n");
scanf("%s",str);
printf("%s",strrev(str));
getch();
}
```

strupr( )

This function converts lowercase string to uppercase string. If any of the character, already in uppercase then the function skips that character and converts the remaining character which is in lowercase.

strupr(string)

Example

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char str[10];
clrscr();
printf("Enter a string\n");
scanf("%s",str);
printf("%s",strupr(str));
getch();
}
```

strlwr( )

This function converts uppercase string to lowercase string. If any of the character, already in lowercase then the function skips that character and converts the other remaining character which is in uppercase.

strlwr(string)

Example

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char str[10];
clrscr();
printf("Enter a string\n");
scanf("%s",str);
printf("%s",strlwr(str));
getch();
}
```

Exercise

1. Write a program to read N students name and display them in alphabetical order.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char name[100][15],temp[15];
int i,j,n;
clrscr();
printf("How many students are there\n");
scanf("%d",&n);
for(i=0;i<n;i++)
{
printf("Enter name of student\n");
scanf("%s",name[i]);
}
for(i=0;i<n-1;i++)
{
for(j=i+1;j<n;j++)
{
if(strcmp(name[i],name[j])>0)
{
strcpy(temp,name[i]);
strcpy(name[i],name[j]);
strcpy(name[j],temp);
}
}
}
printf("Names in alphabetical order\n");
for(i=0;i<n;i++)
{
printf("%s\n",name[i]);
}
getch();
}
```

2. Write program to read a line of text and count no of vowel, no of consonant, no of digits and no of spaces.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char str[100];
int spaces=0,vowel=0,digits=0,consonant=0,i;
clrscr();
puts("Enter a text\n");
gets(str);
strlwr(str);
for(i=0;str[i]!='\0';i++)
{
if(str[i]>='a'&&str[i]<='z')
{
if(str[i]=='a'||str[i]=='e'||str[i]=='i'||str[i]=='o'||str[i]=='u')
vowel++;
else
consonant++;
}
if(str[i]>='0'&&str[i]<='9')
digits++;
if(str[i]==32)
spaces++;
}
printf("vowels
      =%d\nconsonant=%d\ndigits=%d\nSpaces=%d",vowel,consonant,digits,spaces);
getch();
}
```

3. Write a program to check whether a given string is palindrome or not.